



Province of the
EASTERN CAPE
EDUCATION

NATIONAL SENIOR CERTIFICATE

GRADE 12

SEPTEMBER 2013

INFORMATION TECHNOLOGY P1

MARKS: 120

TIME: 3 hours



This question paper consists of 12 pages.

INSTRUCTIONS AND INFORMATION

1. This is a three-hour examination. Because of the nature of this examination it is important to note that you will NOT be permitted to leave the examination room before the end of the examination session.
2. You require the files listed below in order to answer the questions. They are EITHER issued on a CD to you, OR the invigilator/educator will tell you where to find them on the hard drive of the workstation you are using OR in which network folder it is. If the files are issued to you on a CD, you need to copy them onto your hard disk.

QUESTION 1

Question1_u.pas
Question1_p.dpr
Question1_u.dfm
School.mdb
ChildrenTb.txt
ParentsTb.txt

QUESTION 2

Question2_u.pas
Question2_p.dpr
Question2_u.dfm
Applications.txt

QUESTION 3

Question3_u.pas
Question3_p.dpr
Question3_u.dfm
Tickets.txt

If a CD containing the files was issued to you, write your surname on the label.

3. Save your work at regular intervals as a precaution against power failures; replace the 'X' with your surname.
4. Read ALL the questions carefully. Do only what is required by the question.
5. During the examination you may use the manuals originally supplied with the hardware and software. You may also use the HELP functions of the software. You may NOT refer to any other resource material.
6. At the end of this examination session you will be required to hand in the CD given to you by the invigilator with your work saved on it, or you must make sure that all your work has been saved on the network as explained to you by the invigilator/educator. Ensure that all files can be read.
7. You also have to hand in printouts of the programming code for all the questions you have done.
8. All printing of programming questions will take place within an hour of the completion of the examination.

SCENARIO

A local playschool has approached you to help them computerise all their administration to simplify the day-to-day running of the school.

QUESTION 1: DATABASE AND DELPHI

The database, **School.mdb**, which contains data related to this topic, has been supplied to you in a folder named **Question 1**.

Two text files have been supplied as well in case you cannot use the database provided. Use the text files named **ChildrenTb.txt** and **ParentsTb.txt** to create your own database named **School** containing two tables named **ChildrenTb** and **ParentsTb**. Change the data types of the fields of the tables to the specifications given below.

The **ChildrenTb** table stores data the children currently enrolled at the playschool. The fields in this table are defined as follows:

ChildrenTb : Table			
	Field Name	Data Type	Description
?	ChildID	Number	
	Name	Text	Name of Child
	Surname	Text	Surname of Child
	DateOfBirth	Date/Time	Child's date of birth
	Gender	Text	M or F
	Allergies	Yes/No	
▶	ExtraActivities	Text	Ball Skills or Cooking is Fun or Art for Kids

The following table is an example of the data contained in the table named **ChildrenTb** in the database named **School.mdb**.

ChildrenTb : Table							
	ChildID	Name	Surname	DateOfBirth	Gender	Allergies	ExtraActivities
▶	1	Jack	Norton	2008/03/10	M	☒	Ball Skills
	2	Sarah	du Plessis	2009/10/29	F	☐	Art for Kids, Cooking is Fun
	3	Ben	Orsmond	2009/07/19	M	☒	Ball Skills
	4	Lily	Bowes	2009/06/25	F	☐	Cooking is Fun
	5	Jocelyn	Forbes	2010/01/04	F	☐	Art for Kids
	6	Adrian	Kuiper	2009/02/14	M	☐	Ball Skills
	7	Anathi	Bodlo	2009/12/24	F	☒	Ball Skills
	8	Anita	Kleinhans	2009/09/19	F	☐	Cooking is Fun
	9	Ashwin	Willemse	2008/08/11	M	☐	Ball Skills
	10	Peter	Grant	2009/03/20	M	☒	Art for Kids
	11	Chloe	Randall	2009/10/15	F	☐	Ball Skills
	12	Jeff	Sinclair	2009/11/11	M	☐	Cooking is Fun
	13	Nathi	Xinwa	2009/06/14	M	☒	Ball Skills
	14	Wandile	Langeni	2009/02/24	M	☐	Ball Skills
	15	Kerry	Lourens	2009/09/03	F	☒	Art for Kids
	16	Sethu	Duku	2008/01/01	F	☒	Ball Skills
	17	Caryn	Sirgel	2008/05/18	F	☐	Ball Skills, Cooking is Fun
	18	Nicola	Venter	2009/06/24	F	☒	Cooking is Fun

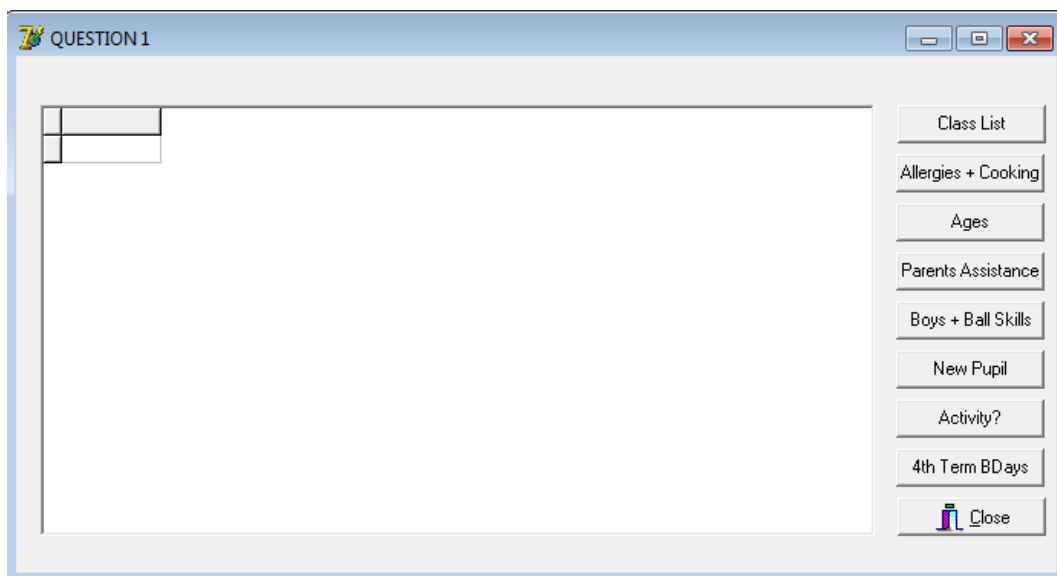
The **ParentsTb** table stores data about the parents of the children at the playschool. The fields in this table are defined as follows:

ParentsTb : Table		
	Field Name	Data Type
	ChildID	Number
	Dad_Name	Text
	Mom_Name	Text
	AvailableToHelp	Yes/No

The following table is an example of the data contained in the table named **ParentsTb** in the database named **School.mdb**.

ParentsTb : Table				
	ChildID	Dad_Name	Mom_Name	AvailableToHelp
▶	1	Shaun	Jessica	☒
	2	Craig	Sandra	☒
	3	Justin	Melanie	☐
	4	Jason	Melissa	☐
	5	Derek	Nancy	☒
	6	Zane	Stacey	☒
	7	Zolani	Amanda	☐
	8	Andre	Tonia	☒
	9	George	Liesl	☒
	10	Tyrone	Zelda	☒
	11	Tristan	Tracey	☐
	12	Grant	Caitlin	☐
	13	Mandla	Victoria	☐
	14	Mzukisi	Ziyanda	☐
	15	Doug	Nancy	☒
	16	Sihle	Sinovuyo	☐
	17	Kevin	Christelle	☒

You have also been supplied with an incomplete Delphi program with a unit named **Question1_u** and a project named **Question1_p** in the folder named **Question 1**. Open the incomplete program.



The program should be able to connect to the database named **School.mdb**. When you do QUESTION 1.1 and you find that the connectivity is not in place, use the following steps to establish connection with the database:

- Click on the ADOQuery component named **qryschoool**.
- Click on the Ellipse button (three dots) to the right of the ConnectionString property in the Object Inspector.
- Click on the Build button which takes you to the Data Link Properties dialogue box.
- Select Microsoft Jet 4.0 OLE DB Provider and click on Next.
- The first option on the Connection tab sheet allows you to browse and find the **School.mdb** file.
- Remove the user name Admin.
- Click on the Test Connection button.
- Click OK on each one of the open dialogue windows.

NOTE: If you cannot establish connectivity with the database at all when you execute the program you must still do and submit the programming code for marking.

Marks will be awarded for the programming code that contains the SQL statements in the unit named Question1_u as well as code that makes use of an inputbox as required by the question.

- 1.1 Complete the code in the **CLASS LIST** button by formulating an SQL statement to display all the information of the children in the class to produce a class list, in alphabetical order of the names.

Example of output for first few records:

ChildID	Name	Surname	DateOfBirth	Gender	Allergies	ExtraActivities
6	Adrian	Kuiper	2/14/2009	M	False	Ball Skills
22	Alex	Fergusson	8/12/2009	M	False	Art for Kids
7	Anathi	Bodlo	12/24/2009	F	True	Ball Skills
8	Anita	Kleinhans	9/19/2009	F	False	Cooking is Fun
9	Ashwin	Willemse	8/11/2008	M	False	Ball Skills
3	Ben	Orsmond	7/19/2009	M	True	Ball Skills
17	Caryn	Sirgel	5/18/2008	F	False	Ball Skills, Cooking is Fun
11	Chloe	Randall	10/15/2009	F	False	Ball Skills
1	Jack	Norton	3/10/2008	M	True	Ball Skills
12	Jeff	Sinclair	11/11/2009	M	False	Cooking is Fun
5	Jocelyn	Forbes	1/4/2010	F	False	Art for Kids
15	Kerry	Lourens	9/3/2009	F	True	Art for Kids
21	Kevin	Rezuidenthout	11/29/2009	M	True	Ball Skills, Art for Kids

(3)

- 1.2 Complete the code in the **ALLERGIES + COOKING** button by formulating an SQL statement to display the names of the children who have an allergy and is doing the 'Cooking is Fun' activity?

Example of output:

name
Nicola

(4)

- 1.3 Complete the code in the **AGES** button by formulating an SQL statement that will display the ages of the children. Age is a calculated field which must display the current age of the child.

Example of output for first few records:

name	Age
Jack	5
Sarah	4
Ben	4
Lily	4
Jocelyn	3
Adrian	4
Anathi	4
Anita	4
Ashwin	5

(4)

- 1.4 Complete the code in the **PARENTS ASSISTANCE** button by formulating an SQL statement that will display the child's name, dad and mom's name for all the children in the class whose parents are prepared to be of assistance.

Example of output for first few records:

name	dad_name	mom_name
Jack	Shaun	Jessica
Sarah	Craig	Sandra
Jocelyn	Derek	Nancy
Adrian	Zane	Stacey
Anita	Andre	Tonia
Ashwin	George	Liesl
Peter	Tyrone	Zelda
Kerry	Doug	Nancy
Caryn	Kevin	Christelle
Nicola	Arthur	Ann
Phila	Chulamanco	Okuhle
Lindelwa	Anathi	Zizipho
Kevin	Callum	Antoinette
Alex	Aidan	Natasha

(5)

- 1.5 Complete the code in the **BOYS + BALL SKILLS** button by formulating an SQL statement to calculate how many boys have enrolled to do Ball Skills. Display with a suitable heading.

Example of output:

Boys doing Ball Skills
12

(2)

- 1.6 A new girl has joined the class. Complete the code in the **NEW PUPIL** button by formulating an SQL statement to add the new learners' details to the **ChildrenTB** table. Display the table once all the information has been inserted.

ChildID: 23
 Name: Rebekka
 Surname: White
 DateOfBirth: 2009/04/02
 Gender: F
 Allergies: No
 ExtraActivities: Cooking is Fun

(4)

- 1.7 Complete the code in the **ACTIVITY?** button by formulating an SQL statement to display the name and surname of the children who have signed up for the particular activity as entered using an InputBox.

Example of the output if **Art for Kids** is entered:

name	surname
Sarah	du Plessis
Jocelyn	Forbes
Peter	Grant
Kerry	Lourens
Phila	Njozela
Kevin	Bezuidenhout
Alex	Fergusson

(4)

- 1.8 Complete the code in the **4TH TERM BDAYS** button by formulating an SQL statement to display Name, Surname and DateOfBirth fields of the children who celebrate their birthday in the last term (October, November or December).

Example of the output:

name	surname	dateofbirth
Sarah	du Plessis	10/29/2009
Anathi	Bodlo	12/24/2009
Chloe	Randall	10/15/2009
Jeff	Sinclair	11/11/2009
Kevin	Bezuidenhout	11/29/2009

(5)

- Enter your name and surname as a comment line in the first line of the file named **Question1_uX.pas** containing the SQL statements.
- Save the unit **Question1_uX** and the project **Question1_pX** (File|Save All).
- Make a printout of the code of the **Question1_uX.pas** file.

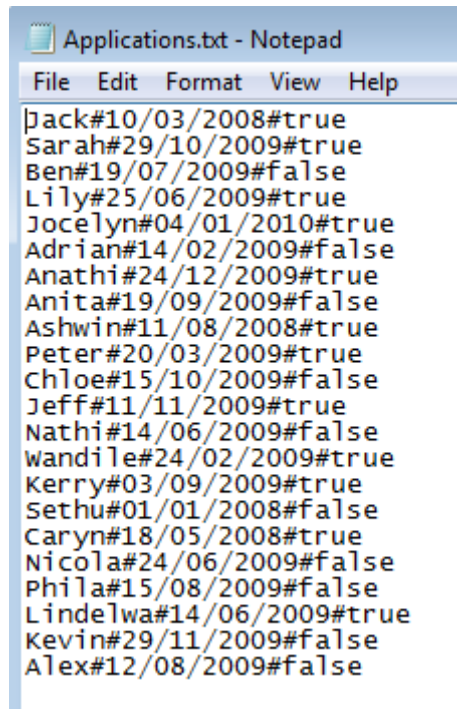
[33]

QUESTION 2: DELPHI PROGRAMMING

This question is intended to test object-oriented programming skills. You are required to produce a solution that includes all classes specified in the instructions. No marks will be allocated to any alternative solution such as one program not creating an object.

You have been supplied with a text file named **Applications.txt**, which contains information of applicants applying to preschools.

The contents of the text file is shown below:



```
Applications.txt - Notepad
File Edit Format View Help
Jack#10/03/2008#true
Sarah#29/10/2009#true
Ben#19/07/2009#false
Lily#25/06/2009#true
Jocelyn#04/01/2010#true
Adrian#14/02/2009#false
Anathi#24/12/2009#true
Anita#19/09/2009#false
Ashwin#11/08/2008#true
Peter#20/03/2009#true
Chloe#15/10/2009#false
Jeff#11/11/2009#true
Nathi#14/06/2009#false
Wandile#24/02/2009#true
Kerry#03/09/2009#true
Sethu#01/01/2008#false
Caryn#18/05/2008#true
Nicola#24/06/2009#false
Phila#15/08/2009#false
Lindelwa#14/06/2009#true
Kevin#29/11/2009#false
Alex#12/08/2009#false
```

The name of the child is given first, the date of birth followed by whether the deposit has been paid or not.

2.1 Create an object class (another unit) named **Applications_uX** and save this unit as **Applications_uX** in your **Question2_X** folder (X should be replaced by your surname). Write the following code as part of this class:

2.1.1 Define a class named **TSchool**. This class must contain the following private fields:

fname
fdob
fdeposit
faccept
fage

2.1.2 Ensure that you choose appropriate data types for these fields. (3)
Write a parameterised **Constructor** method which accepts the name, date of birth and whether the deposit has been paid. All the fields must be initialised in the constructor. (2)

- 2.1.3 Write a method **CalcAge** which will calculate the current age of each applicant. (3)
- 2.1.4 Write a boolean method named **Accepted** which will determine whether an applicant has been accepted. To be accepted the applicant must be between 4 and 6 years old (both inclusive) and the deposit must be paid. (4)
- 2.1.5 Write a method named **GetReason** which will return the reason why an applicant was not accepted. Refer to QUESTION 2.1.4 to see conditions for acceptance. (7)
- 2.1.6 Write an appropriately named 'get' method (accessor method) to return the name of the applicant. (2)
- 2.1.7 Write an appropriately named 'get' method (accessor method) to return the date of birth of the applicant. (2)
- 2.2 Write code to do the following in the **Question2_uX** unit in the given program.
- 2.2.1 Create an array named **arrSchools** that holds objects of **TSchool**. Write code in the OnActivate Event Handler of the form to read information from the text file **Applications.txt** according to the following steps:
- (a) Test if the file exists. Display a suitable message if the file does not exist and terminate the program.
 - (b) Use a loop to:
 - Read a line of text from the text file
 - Separate the text into the name, date of birth and whether deposit has been paid (boolean)
 - Create a new **TSchool** object and place the object in the array named **arrSchools**.
 - (c) Use a counter variable to keep track of how many items there are in the array. (14)

- 2.2.2 **DISPLAY ALL APPLICANTS:** Display the names of all the applicants that are applying for preschool.

```
Jocelyn
Adrian
Anathi
Anita
Ashwin
Peter
Chloe
Jeff
Nathi
Wandile
Kerry
Sethu
Caryn
Nicola
Phila
Lindelwa
Kevin
Alex
```

(2)

- 2.2.3 **ACCEPTED CANDIDATES:** Display the names of the applicants who have been accepted. Also display how many applicants have been accepted.

```
Jack
Sarah
Lily
Anathi
Ashwin
Peter
Jeff
Wandile
Kerry
Caryn
Lindelwa
11 children accepted
|
```

(6)

- 2.2.4 **REASONS FOR NON-ACCEPTANCE:** If the applicant was not accepted, display the applicant's name as well as the reason why he/she was not accepted. Reasons could be one of the following: Incorrect Age; Deposit has not been paid.

```
Ben      Deposit not paid
Jocelyn  Incorrect Age
Adrian   Deposit not paid
Anita    Deposit not paid
Chloe    Deposit not paid
Nathi    Deposit not paid
Sethu    Deposit not paid
Nicola   Deposit not paid
Phila    Deposit not paid
Kevin    Deposit not paid
Alex     Deposit not paid
```

(5)

- Enter your name and surname as a comment line in the first line of the files named **Question2_uX.pas** and **Applications_uX.pas**.
- Save the units **Question2_uX**, **Applications_uX** and the project **Question2_pX** (File|Save All).
- Make a printout of the code of the **Question2_uX.pas** as well as the **Applications_uX.pas** file.

[50]

QUESTION 3: DELPHI PROGRAMMING

The school is planning an event to raise money for new equipment and also to donate some of the money to a less privileged school. Tickets are sold by the parents and there will also be lucky draw prizes.

The names of the children are stored as a constant in the array named **arrnames**. There are 23 children in the class. Code has been completed to display the names in the StringGrid component.

NOTE: The output for the lucky draw winners will differ from the example, as it is randomly generated.

3.1 GET INFO:

Write code to retrieve the ticket numbers that have been sold by the children. The information is stored in a text file named **tickets.txt**. Store all the information in a two-dimensional array named **ar2tickets**.

(9)

3.2 DISPLAY:

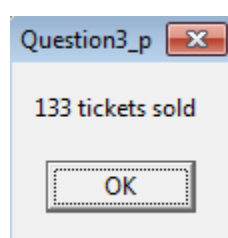
Display the ticket numbers each child sold, in the StringGrid.

Jack	1	5	65	3	87	56	45	127	0	0
Sarah	43	2	27	60	61	126	0	0	0	0
Ben	4	26	29	85	102	164	0	0	0	0
Lily	23	28	58	59	121	0	0	0	0	0
Jocelyn	7	22	57	83	84	161	0	0	0	0
Adrian	25	30	62	100	128	0	0	0	0	0
Anathi	6	21	82	99	120	171	0	0	0	0
Anita	31	63	64	118	158	203	0	0	0	0
Ashwin	20	32	81	97	132	0	0	0	0	0
Peter	10	55	86	103	155	0	0	0	0	0
Chloe	33	67	79	80	144	0	0	0	0	0
Jeff	19	54	70	110	136	0	0	0	0	0
Nathi	8	34	78	35	139	0	0	0	0	0
Wandile	9	51	52	53	116	174	175	176	184	0
Kerry	36	71	72	95	140	0	0	0	0	0
Sethu	17	73	87	106	154	0	0	0	0	0

(3)

3.3 TICKETS SOLD?

Write code to calculate how many tickets were sold. Display the output making use of a ShowMessage component.



(5)

3.4 LUCKY DRAWS

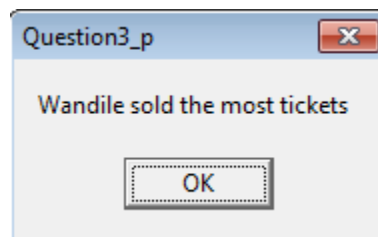
Using the total calculated in QUESTION 3.3; get 10 random numbers (between 1 and 230) in order to award the 10 lucky draw prizes. If the random number does not represent a ticket that was sold then a new random number must be generated.

Jack	1	5	65	3	87	56	45	127	0	0
Sarah	43	2	27	60	61	126	0	0	0	0
Ben	4	26	29	85	102	164	0	0	0	0
Lily	23	28	58	59	121	0	0	0	0	0
Jocelyn	7	22	57	83	84	161	0	0	0	0
Adrian	25	30	62	100	128	0	0	0	0	0
Anathi	6	21	82	99	120	171	0	0	0	0
Anita	31	63	64	118	158	203	0	0	0	0
Ashwin	20	32	81	97	132	0	0	0	0	0
Peter	10	55	86	103	155	0	0	0	0	0
Chloe	33	67	79	80	144	0	0	0	0	0
Jeff	19	54	70	110	136	0	0	0	0	0
Nathi	8	34	78	35	139	0	0	0	0	0
Wandle	9	51	52	53	116	174	175	176	184	0
Kerry	36	71	72	95	140	0	0	0	0	0
Sethu	17	73	87	106	154	0	0	0	0	0

(6)

3.5 MOST SOLD

Calculate which child sold the most tickets and display the child's name using a ShowMessage component.



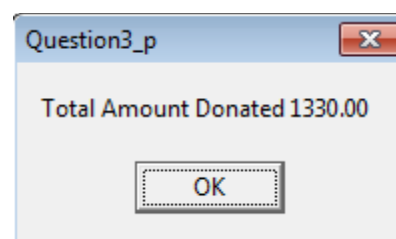
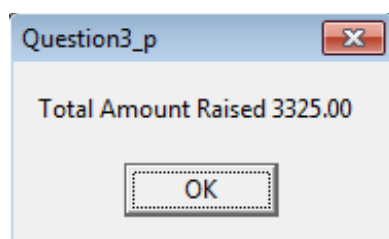
(9)

3.6 AMOUNT RAISED?

Calculate and display, making use of ShowMessage components:

- the amount of money raised if each ticket cost R25.
- 40% of the total amount was donated to an underprivileged school. Display the amount that was donated.

Both amounts must be rounded off to two decimal places.



(5)

- Enter your name and surname as a comment line in the first line of the file named **Question3_uX.pas**.
- Save the unit **Question3_uX** and the project **Question3_pX** (File|Save All).
- Make a printout of the code of the **Question3_uX.pas** file.

[37]

TOTAL: 120